# Learning VEX and the New Wrangle SOPs

## A Gentle Introduction

**Ari Danesh**
ari@sidefx.com

SIDE EFFECTS SOFTWARE

# Objectives

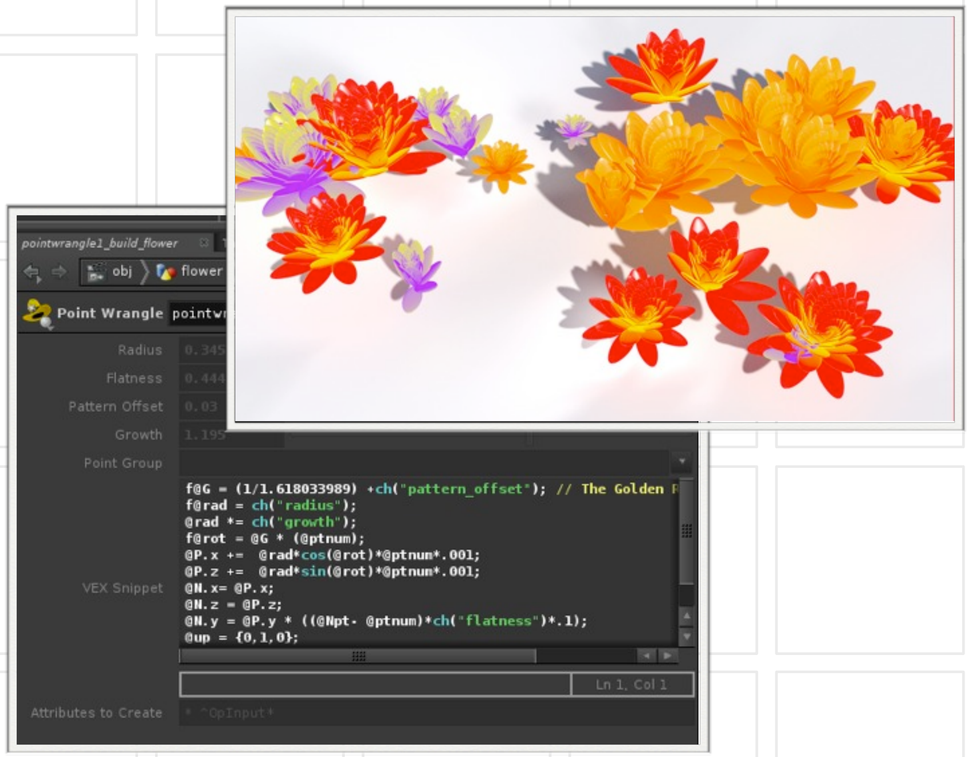
Understand the Syntax of VEX for the Wrangle Nodes

Understand the Differences between VEX and Wrangle Node code writing

Start to think how you can incorporate Wrangle Nodes and VEX into your work flow

SIDE EFFECTS SOFTWARE

# Agenda




- What are the Wrangle Nodes?
- What is VEX?
- Why use VEX and Wrangle Nodes?
- Wrangle Syntax
- A Practical Wrangle Example
- Creating a VEX Type?
- VEX in Different Contexts

SIDE EFFECTS SOFTWARE

# Requirements

Need to have Houdini 12.5 or greater installed

Wrangle Nodes are new to 12.5.

Will work with Houdini, HoudiniFX, and Apprentice

SIDE EFFECTS SOFTWARE

# Wrangle Nodes

Just Awesome!

SIDE EFFECTS SOFTWARE

# Definition from Merriam Webster

WRANGLE - from http://www.merriam-webster.com/dictionary/wrangle

intransitive verb

1: to dispute angrily or peevishly : bicker

2: to engage in argument or controversy

transitive verb

1: to obtain by persistent arguing or maneuvering : wangle

2[back-formation from wrangler] : to herd and care for (livestock and especially horses) on the range

**This is the one I like :)**

SIDE EFFECTS SOFTWARE

# Definition from Ari

**to herd and care for (points, attributes, and volumes) in Houdini**

SIDE EFFECTS SOFTWARE

# Wrangle Nodes

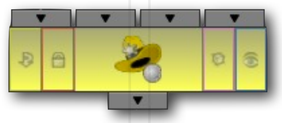
New “wrangle” nodes let you write short VEX snippets to manipulate geometry.

Point Wrangle - lets you manipulate point attributes in a SOP context.

Attribute Wrangle - lets you manipulate any geometry attribute in a CVEX context.

Volume Wrangle - lets you change voxel values.

SIDE EFFECTS SOFTWARE



# Point Wrangle

**Warning!**
This node requires that you understand the vex language. It is very easy to write incorrect code using this node.

Runs a VEX snippet to modify point attributes, including position.

This node corresponds to the VOP SOP, but uses a textual VEX snippet instead of a VOP network.

This node runs the snippet on every point in the input geometry. The snippet can edit the input geometry by changing attributes. It can access information from other geometry using attributes and VEX functions.

SIDE EFFECTS SOFTWARE

# VEX Definition

SIDE EFFECTS SOFTWARE

# What is VEX?

**VEX** is **not** a general purpose scripting language like HScript or Python. Specialized for Shader Writing and Node Creation!

VEX - Vector Expression

VEX - Like its visual equivalent “VOPS” is a SIMD Language

VEX is extremely fast. Performance is similar to Compiled in C/C++ code. In some contexts can exceed C/C++ (fur)

Syntax - Similar to C/C++ put also takes cues from RSL (Renderman Shading Language)

SIDE EFFECTS SOFTWARE

# VEX is Used Throughout Houdini

**Modeling - VEX SOP allows you to write a custom surface node that manipulates point attributes**

- Move Points
- Change Colors
- Adjust Velocities

**VEX** or **Point Wrangle** SOP can execute 10 times faster then the **Point SOP**

SIDE EFFECTS SOFTWARE

# VEX is Used Throughout Houdini

**Rendering - Mantra uses VEX for all shading computation**

- lights
- surface
- displacement

**CHOPs**

- Create Custom CHOPs
- VEX code can run faster then compiled code

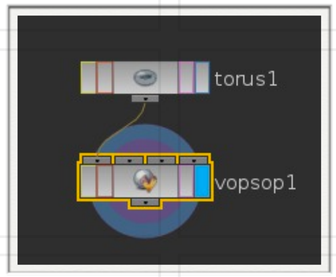
SIDE EFFECTS SOFTWARE

# What Does VEX Look Like?

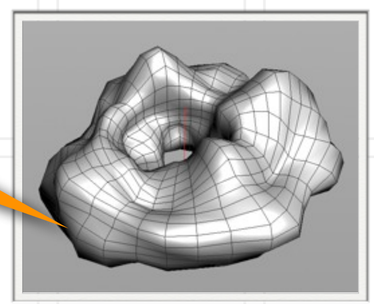
**Since VOPs generates VEX code. Let us experiments by dropping down a VOPSOP**

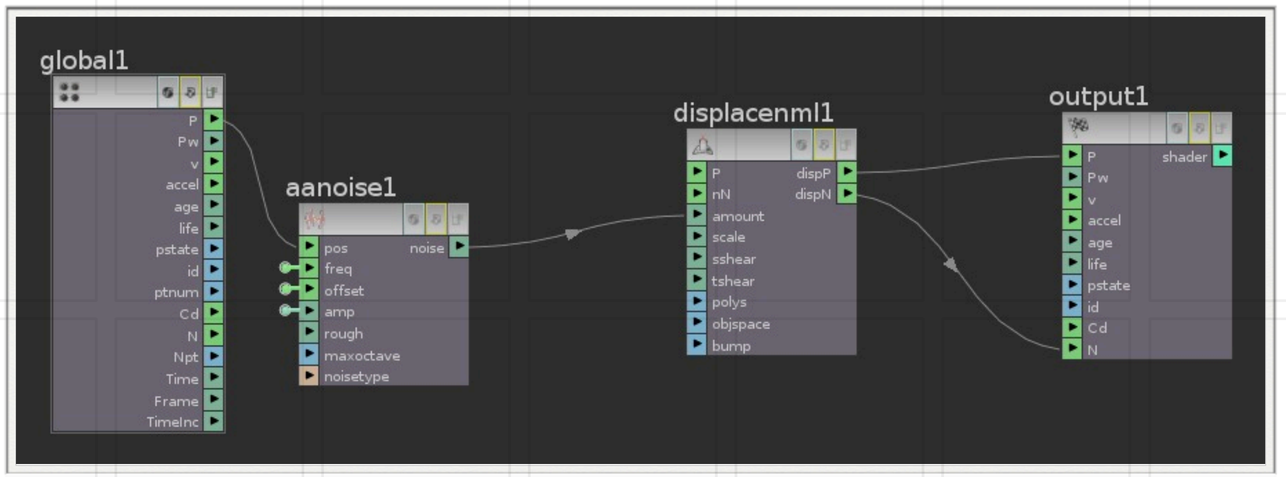
**In a new project**

- Drop down a Torus - Dive inside the Geometry
  - Make it a NURB (Rows, Cols = 30,30)
  - Append a VOPSOP - Dive Inside
  - Create the Network show below




End Result






SIDE EFFECTS SOFTWARE

# What Does VEX Look Like? (cont.)

## Go back to the VOPSOP Level

- Right click on the VOPSOP
- Select View VEX CODE...

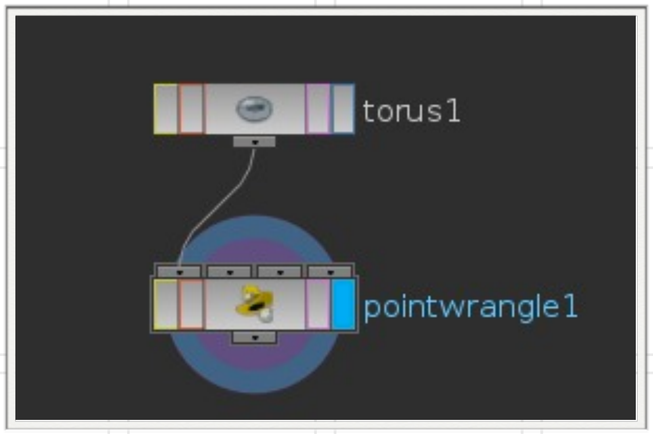

SIDE EFFECTS SOFTWARE

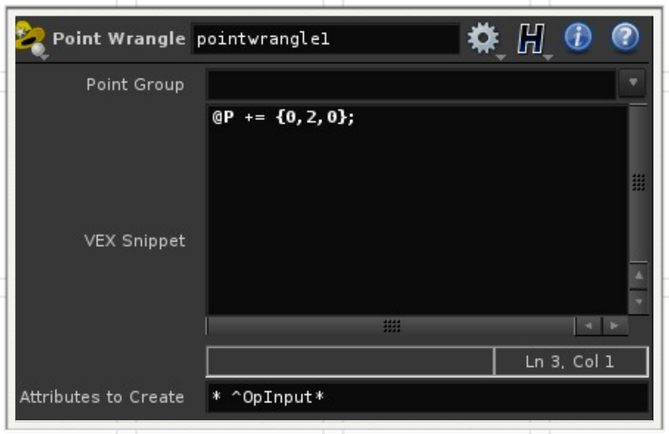
# Back to the Point Wrangle Node

Syntax and Help

SIDE EFFECTS SOFTWARE

# Syntax




- New Project
- Drop down another Torus
- Dive inside
- Append a Point Wrangle
- Type
  - @P += {0,2,0};

SIDE EFFECTS SOFTWARE

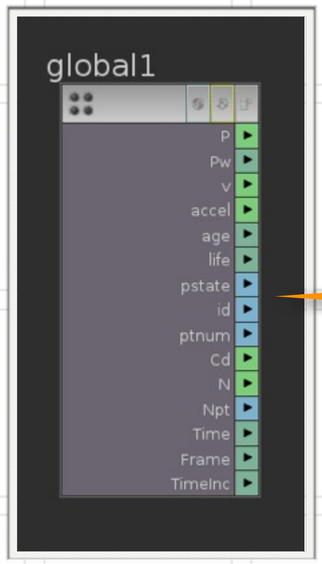
# What does @P mean?

**The “@” symbol means in other languages “to fetch”**

- In our case we are “fetching” or “setting” the attribute “P” (position)

**How do I know to use “P” for Position?**

- Remember VOPS is a visual representation of VEX
- Therefore attributes in VOPS translate to VEX




**P, v, accel, ptnum, and all the others work in the Point Wrangle Node**

SIDE EFFECTS SOFTWARE

**P += {0,2,0};**

**In simple terms - Add to the Point Position (of every point) the Vector 0,2,0**

**P - Point Attribute**

**+= short hand for “add  to the current value”**

- long version “@P = @P + {0,2,0};

**; - every line must end with a “;” (there a few exceptions including conditionals and iterators”)**

- The semicolon means “end of statement”

SIDE EFFECTS SOFTWARE

# Operators

| Arithmetic | |
|---|---|
| + | addition |
| - | subtraction |
| = | “=” - gets assigned the value of |
| + = | Add to the current value |
| - = | Subtract from the current value |
| * = | Multiply to the current value |
| / = | Divide from the current value |
| / | division |
| % | modulus |
| + + | add one |
| -- | subtract one |

| Conditionals | |
|---|---|
| < | less |
| < = | less then or equal to |
| > | greater then |
| > = | greater then or equal to |
| = = | is equal to |
| ! = | is not equal to |
| ! | logical negation |
| ~ | ones complement |
| && | logical AND |
| \|\| | Logical OR |
| | |
| | |
| | |

SIDE EFFECTS SOFTWARE

# {...} is NOT Efficient

Instead of using {...} use the “set()” funtion

To make the Wrangle nodes as efficient as possible match data types for the compiler

Replace P += {0,2,0}; with

- v@P1 = set(0,2, 0);
- @P += @P1;

**What is the “v” in v@P1?**

SIDE EFFECTS SOFTWARE

# Converting Data Types for the Wrangle Node

| VEX | Wrangle | Example |
|---|---|---|
| int | i | i@foo = 4 |
| float | f | f@foo = 4.0 |
| vector | 3@ | v@foo = {32.5, 17.9,-22.3} |
| vector4 | 4@ | v4@foo = {0.5,0.3, 0,0, 0.2} |
| | | |
| | | |
| | | |
| | | |
| string | s@ | s@foo =“hello world” |
| | | |

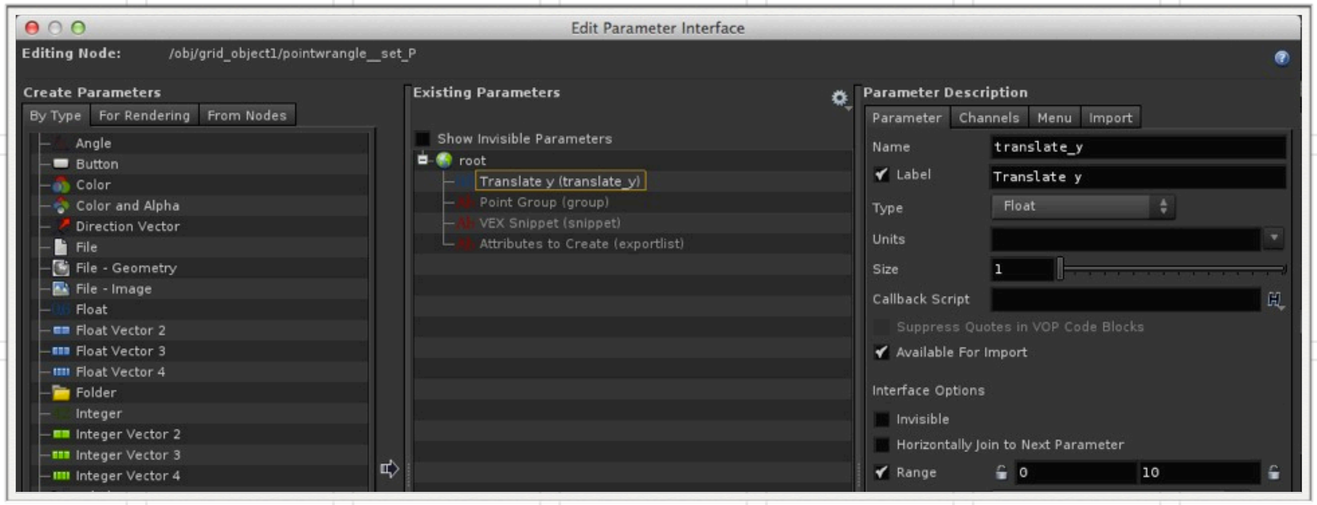
SIDE EFFECTS SOFTWARE

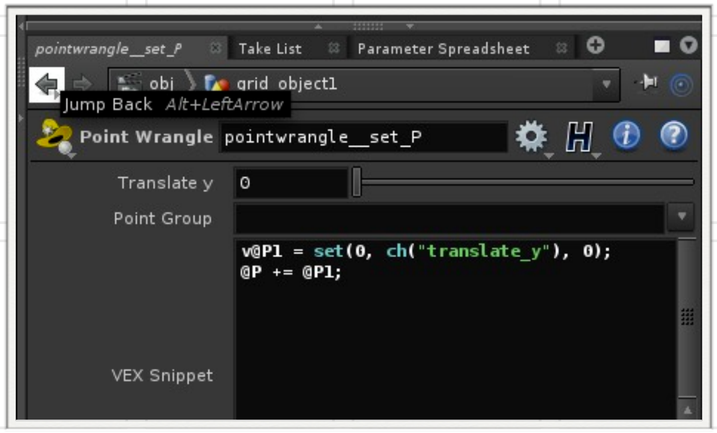
# Data Types in VEX

| Type | Definition | Example |
|---|---|---|
| int | integer values | 21, -3, 7 |
| float | floating point scalar values | 21.3, -3.2, 1.0 |
| vector | Three Floating Point Values | {32.5, 17.9,-22.3} - (e.g., P, v, rgb) |
| vector4 | Four Floating Point Values | {0.5,0.3, 0,0, 0.2} - (e.g., rgba) |
| array | list of values | {3,1,4,5,8,7,3,5,1,0} |
| struct | A fixed set of named values | |
| matrix | Sixteen floating point values representing a 3D Transform Matrix | {{1,0,0,0}, {0,0,1,0}, {0,0,1,0}, {0,0,0,1}} |
| matrix3 | Nine floating point values representing a 2D Transform Matrix or 3D Rotation | |
| string | A string of characters | “hello world” |
| bsdf | bidirectional scattering distribution function | |

SIDE EFFECTS SOFTWARE

# Adding a User Interface

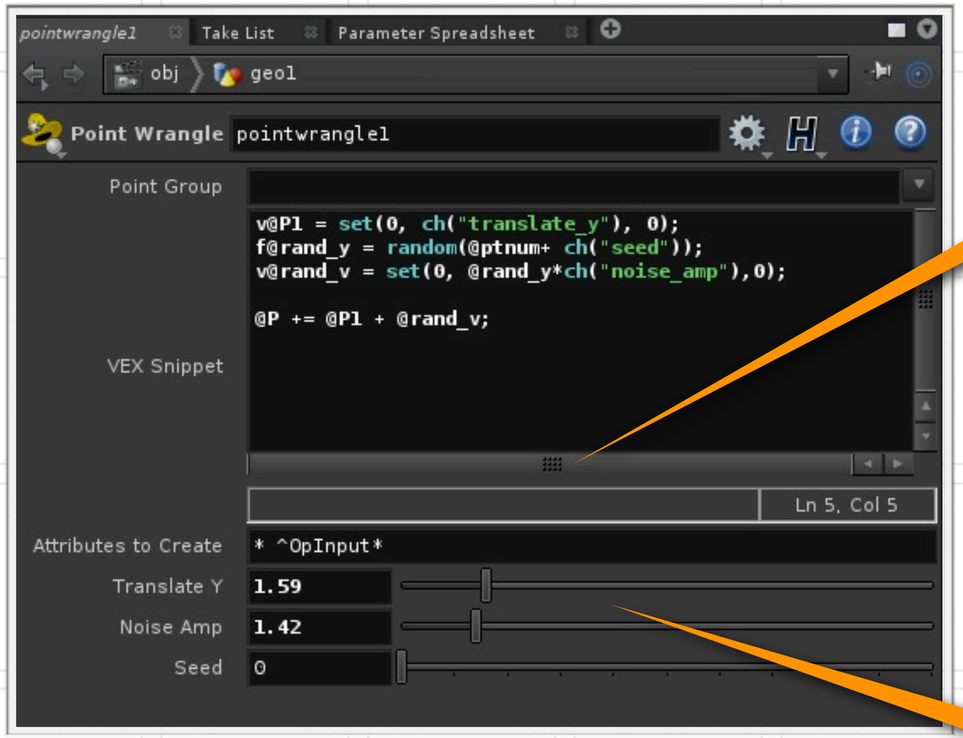



- Edit Parameter Interface
  - name: translate_y
  - label: Translate y
  - Range : 0-10
- Replace v@P1 = set(0,2, 0);
  - with v@P1 = set(0,ch(“translate_y”), 0)

SIDE EFFECTS SOFTWARE

# Random Function on Grid



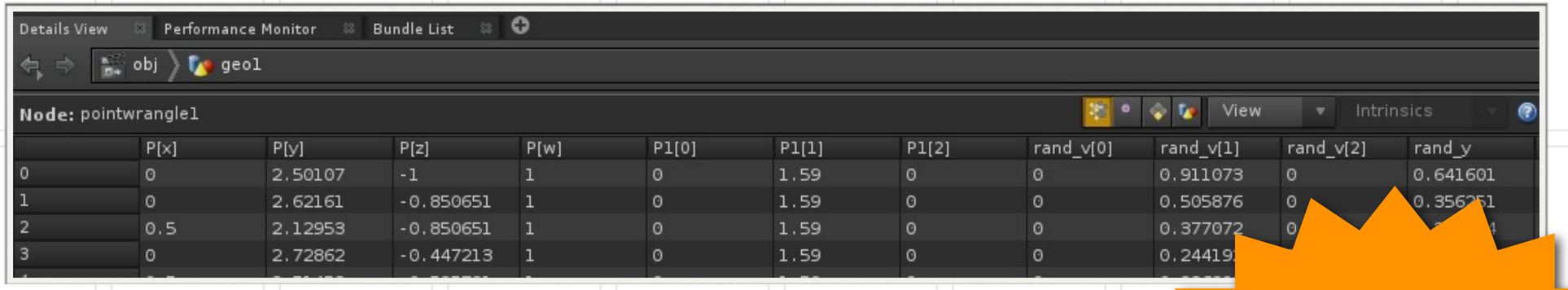

**Why "ptnum" instead of "$PT"**

**Why "random()" instead of "rand()?"**

- Drop down a Grid
  - Rows, Cols (50,50)
- Append a Point Wrangle

SIDE EFFECTS SOFTWARE

# Open up the Details View




| | P[x] | P[y] | P[z] | P[w] | P1[0] | P1[1] | P1[2] | rand_v[0] | rand_v[1] | rand_v[2] | rand_y |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 2.50107 | -1 | 1 | 0 | 1.59 | 0 | 0 | 0.911073 | 0 | 0.641601 |
| 1 | 0 | 2.62161 | -0.850651 | 1 | 0 | 1.59 | 0 | 0 | 0.505876 | 0 | 0.356251 |
| 2 | 0.5 | 2.12953 | -0.850651 | 1 | 0 | 1.59 | 0 | 0 | 0.377072 | 0 | |
| 3 | 0 | 2.72862 | -0.447213 | 1 | 0 | 1.59 | 0 | 0 | 0.24419 | | |

- Notice every variable you created has automatically generated an Attribute
- This is awesome! Instead of slowly creating attributes with ATTRIBCREATE we can use a POINT WRANGLE to quickly generate multiple attributes

**While it is great we can quickly create attributes we do not want to weigh down our geometry with attributes that are not needed (waste of memory)**

SIDE EFFECTS SOFTWARE

# Creating Temporary Variables

If you want to create variables that will not generate attributes you can create a temporary variable

The syntax of a temporary variable is slightly different. As an example instead of type v@foo = {3.0, 4.0, 5.0};

- vector foo = {3.0, 4.0, 5.0};

Other examples

- float bar = 5.0
- int seed = 4;

SIDE EFFECTS SOFTWARE

# Random function on Grid  Re-Written with Temp Variables

```
vector P1 = set(0, ch("translate_y"), 0);

float rand_y = random(@ptnum+ ch("seed"));

// maybe I want rand_v as an attribute

v@rand_v = set(0, rand_y*ch("noise_amp"),0);

@P += P1 + @rand_v;
```

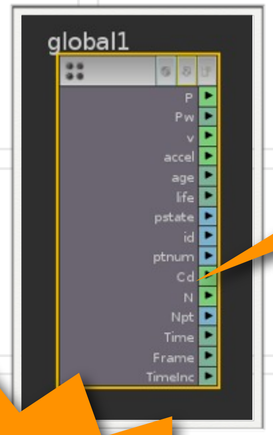
SIDE EFFECTS SOFTWARE

# Two Ways of Commenting Your Code

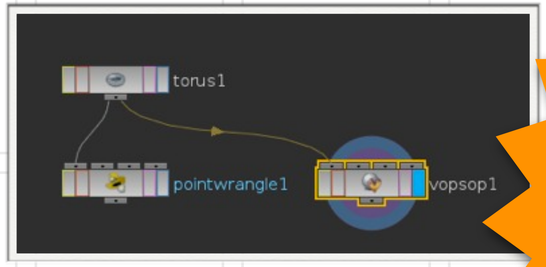
```
// If you want to make a comment for a single line start the line with “//”
/* if you want to make a longer block comment that spans
   several lines. The you need to use the /* system.
   to do that start the comment with “/*” and end it with “*/”
*/
```

SIDE EFFECTS SOFTWARE

# How do we know what the names of Attributes are?




Legal Names




**In VEX you can not use Local Variables.**

**As an Example $PT is not legal**

**Remember names in VEX are context sensitive and case sensitive**

**Just because the name is valid for a SOP does not mean you can use it in a COP**

- Two ways of looking up legal names of attributes
  - Drop down a VOPSSOP
    - Dive Inside
    - The names of the Global Variables are the names you can use in a Point Wrangle

SIDE EFFECTS SOFTWARE

# One more way to look up Global Variable Names

- Open up a terminal/shell window
- Type the command “vcc -X sop”
- vcc -X [ --list-context ] <context>
  - Print out global variables and function signatures for a given context. An argument value of 'contexts' can be used to list which contexts are available.

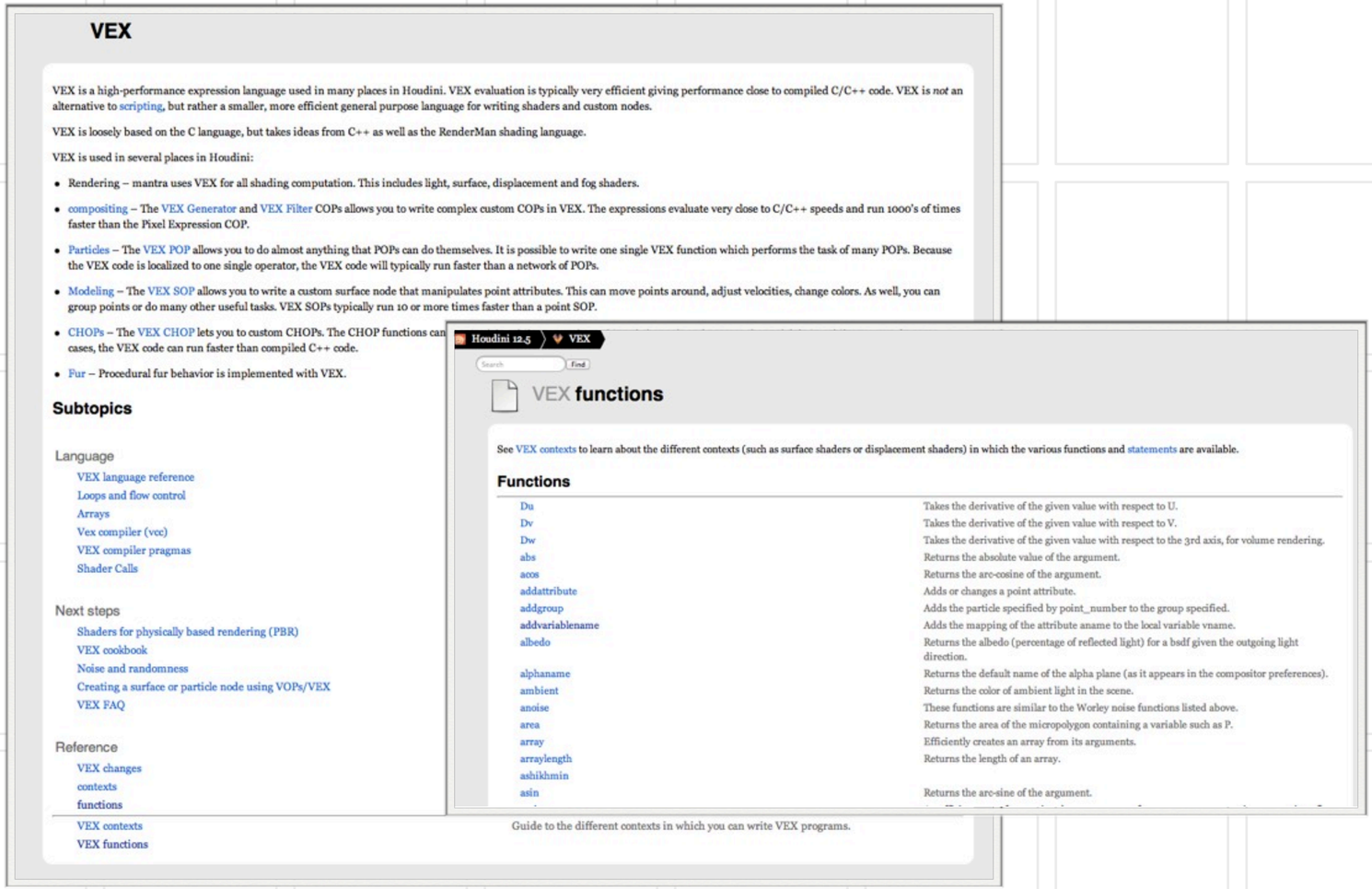
```
Oriel:fibFlower ari$
Oriel:fibFlower ari$
Oriel:fibFlower ari$ vcc -X SOP
Context: Sop

Global Variables:
  (Read/Write)  vector P
  (Read/Write)   float Pw
  (Read/Write)  vector v
  (Read/Write)  vector accel
  (Read/Write)   float age
  (Read/Write)   float life
  (Read/Write)     int pstate
  (Read/Write)     int id
  (Read Only)      int ptnum
  (Read/Write)  vector Cd
  (Read/Write)  vector N
  (Read Only)      int Npt
  (Read Only)    float Time
  (Read Only)    float Frame
  (Read Only)    float TimeInc
```

SIDE EFFECTS SOFTWARE

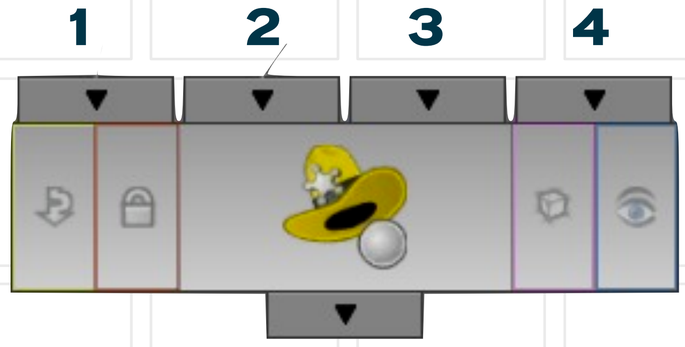
# Why random() instead of rand()

- Answer: This is VEX not HScript
- To lookup a VEX function
  - Open Help
    - Search for VEX
    - In the VEX page click on the functions link




SIDE EFFECTS SOFTWARE

# Using Multiple Inputs in the Wrangle Node




- To get the value of a specific input context use the variable “OpInputN”
- As an example to get the second input use:
  - @OpInput2

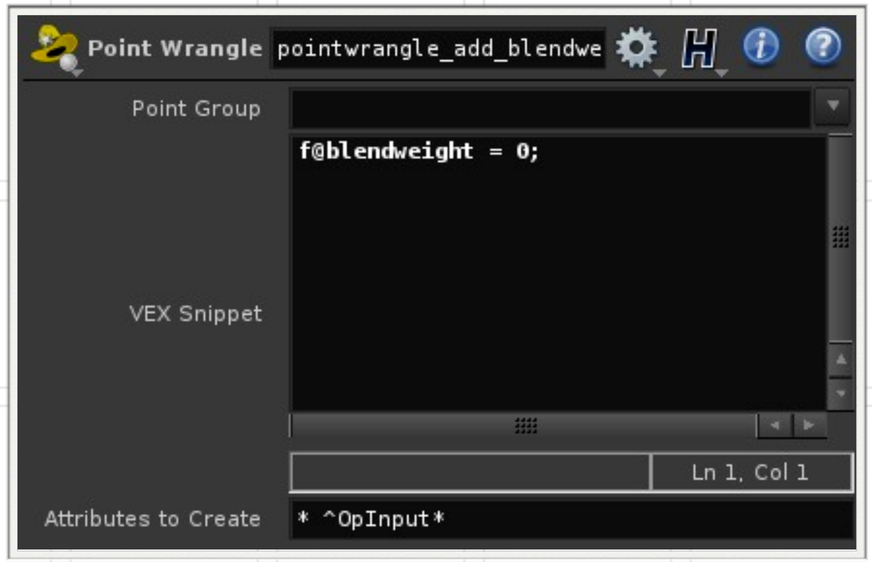
SIDE EFFECTS SOFTWARE

# Example - Blending Between a Grid and a Mountain

SIDE EFFECTS SOFTWARE

# The Network

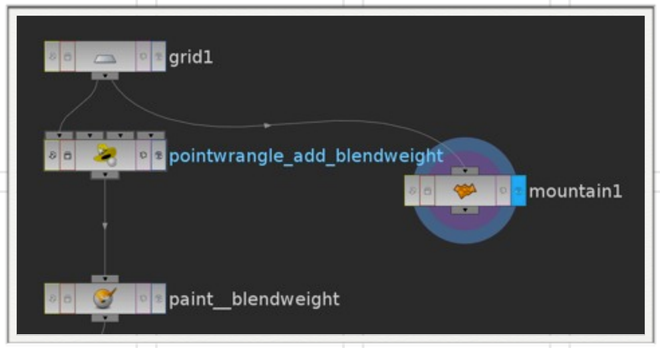
- Drop down a Grid (rows, cols = 50,50)
- Create an attribute called “blend weight”
  - Instead of using the “AttribCreate” use a PointWrangle
  - f@blendWeight = 0;

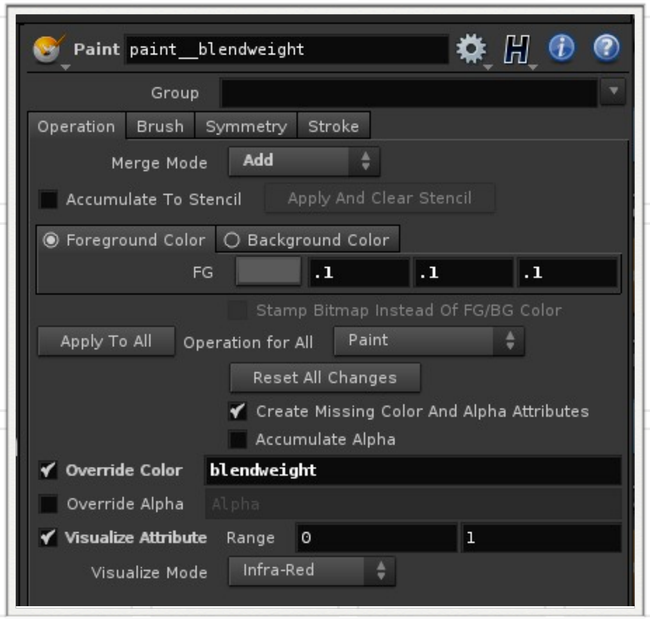

**If the variable has not been declared before the first time you use is you must specify what data type it is**
**f@ - variable is a float value**

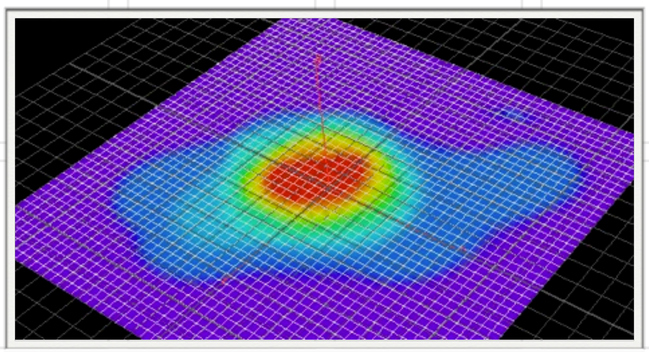
SIDE EFFECTS SOFTWARE

# The Network (cont.)



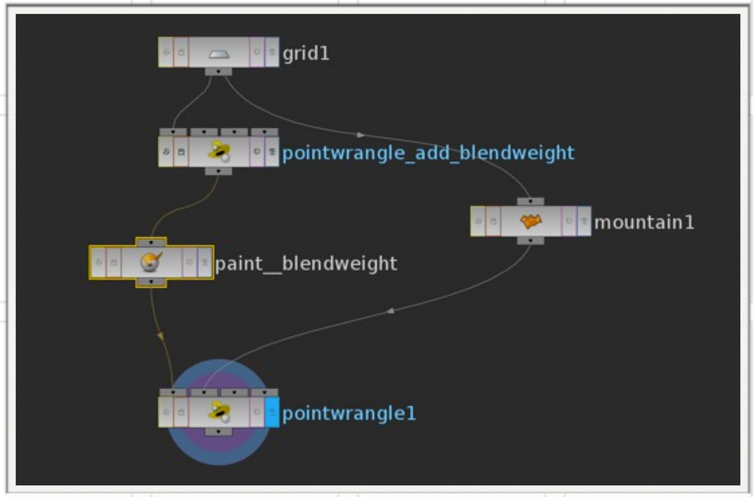

- From the Grid Append a Mountain SOP and add some noise and amplitude
- Append a Paint SOP to the Grid
  - Override Color - blendweight
  - Paint some color on the Grid to blend in some “blendweight”



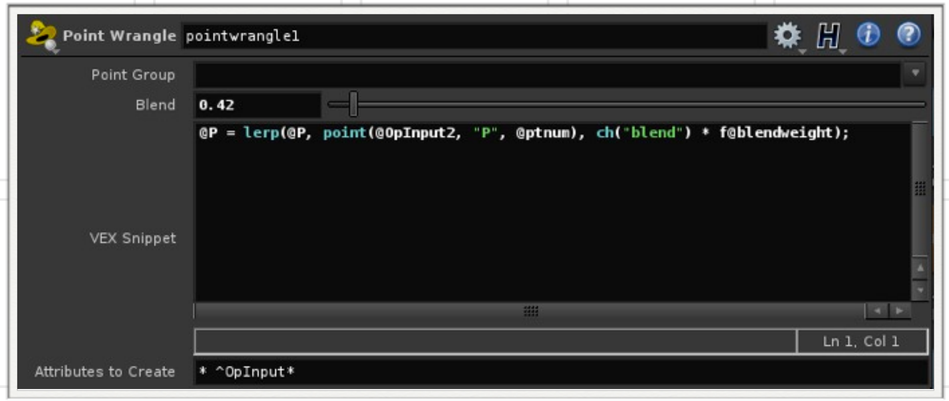



SIDE EFFECTS SOFTWARE

# The Network (cont.)




- Append another Point Wrangle
  - Place the Paint SOP  as Context 1
  - The Mountain SOP as Context 2
  - Paint some color on the Grid to blend in some “blendweight”
  - In the Point Wrangle add a UI for Blend
    - float (name - blend, label - Blend)




- @P = lerp(@P, point(@OpInput2, "P", @ptnum), ch("blend") * f@blendweight);

SIDE EFFECTS SOFTWARE

# Understanding the Expression

- @P = lerp(@P, point(@OpInput2, "P", @ptnum), ch("blend") * f@blendweight);
- **lerp()**
  - Performs bilinear interpolation between the values. If the amount is outside the range 0 to 1, the values will be extrapolated linearly.
  - Performs bilinear interpolation between the values.
  - float lerp(float value1, float value2, float amount)
  - vector lerp(vector value1, vector value2, float amount)
  - vector4 lerp(vector4 value1, vector4 value2, float amount)

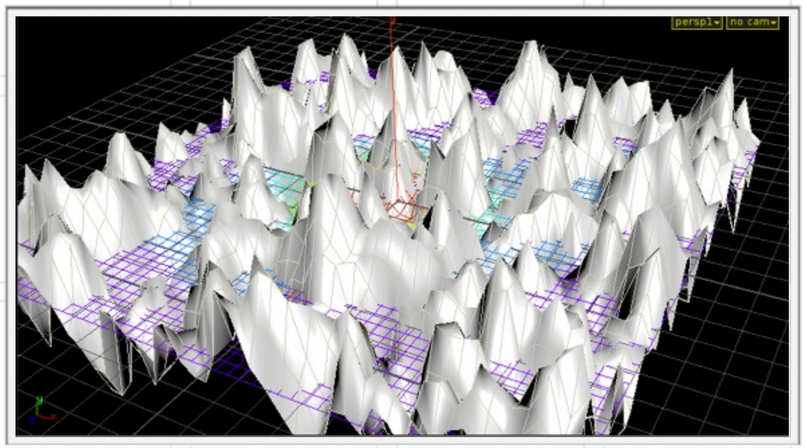
SIDE EFFECTS SOFTWARE

# Understanding the Expression (cont.)

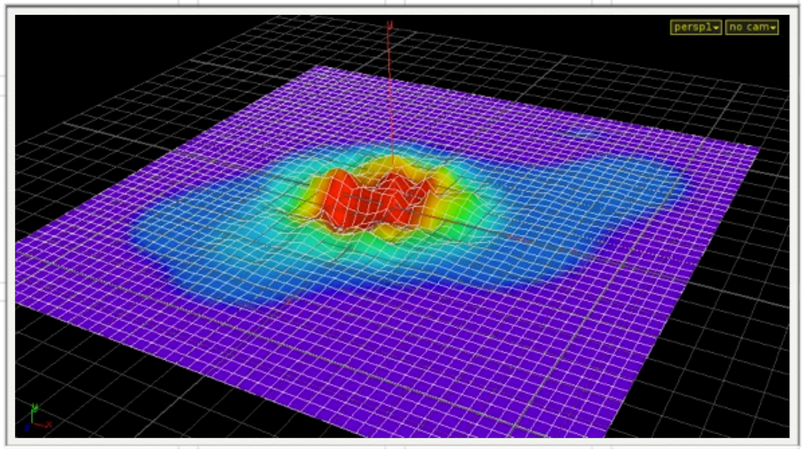
- @P = lerp(@P, point(@OpInput2, "P", @ptnum), ch("blend") * f@blendweight);
- **point()**
  - type point(string geometry, string attribute_name, int pointnumber)
  - Returns 0 if importing the attribute failed, the value of the attribute on success.
  - geometry - The name of the geometry file to reference. attribute name - The name of the attribute (i.e. “Cd” or “P”)
  - number - The point number.
  - Example
    - Get the position of point 3 in "defgeo.bgeo" - pos = point("defgeo.bgeo", "P", 3);

SIDE EFFECTS SOFTWARE

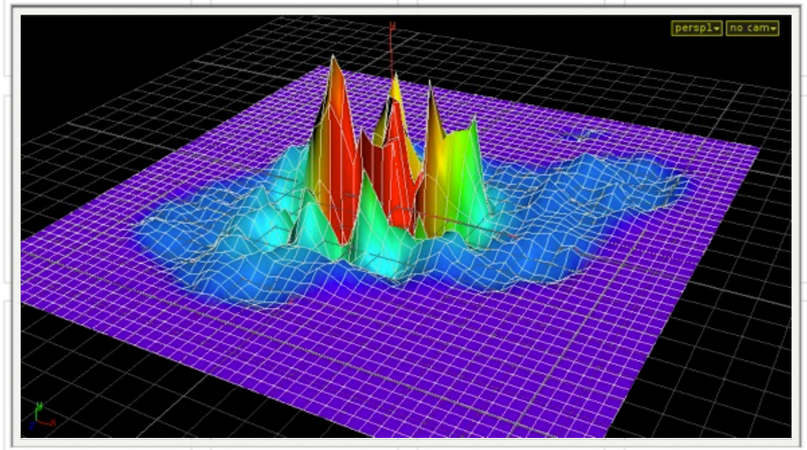
# The Result




Original Mountain




Blend - 0.27




Blend - 2.0

SIDE EFFECTS SOFTWARE

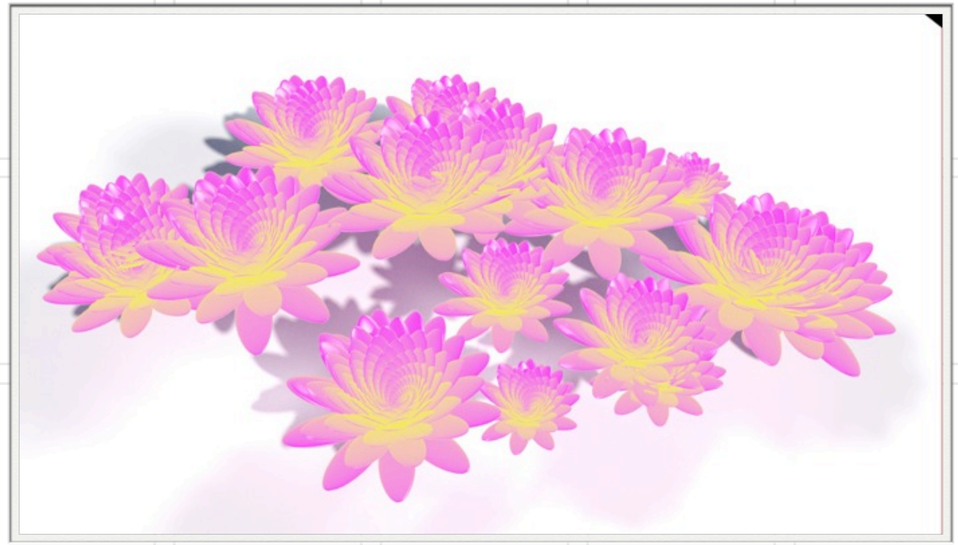
# A More Complex Example

Flower Built on the Golden Ratio

SIDE EFFECTS SOFTWARE

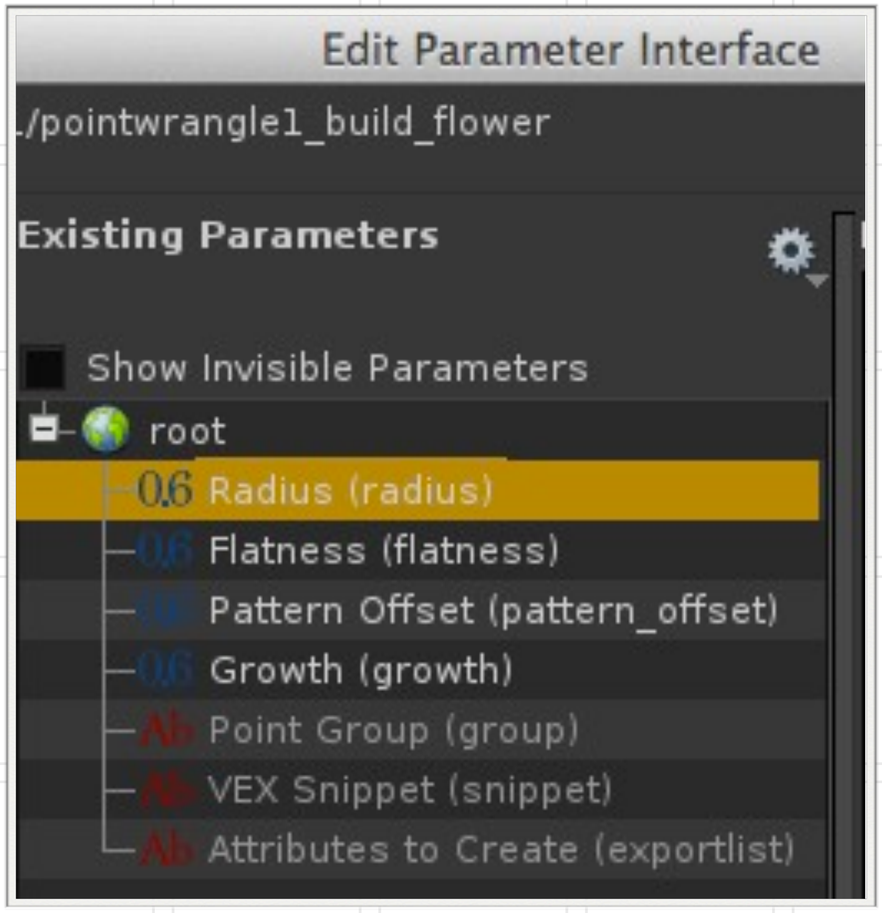
# Objective

- Using the Golden Ratio translate points into the position of a flower
- Add a attribute to the points so older petals are larger then new petals
- Add a attribute to the points to control how open the flowers are
- Control the radius of the flower



SIDE EFFECTS SOFTWARE

# User Interface




SIDE EFFECTS SOFTWARE

# Dot Notation

If you want to set or fetch one component of an attribute you can use dot notation.

As an example if I want to add 3.5 to the z component of position I can:

- @P += {0,0,3.5};
- or
- @P.z += 3.5;

SIDE EFFECTS SOFTWARE

# Code

```
f@G = (1/1.618033989) +ch("pattern_offset"); // The Golden Ratio in Radians
f@rad = ch("radius");
f@rad *= ch("growth");
f@rot = @G * (@ptnum);
@P.x +=  @rad*cos(@rot)*@ptnum*.001;
@P.z +=  @rad*sin(@rot)*@ptnum*.001;
@N.x= @P.x;
@N.z = @P.z;
@N.y = @P.y * ((@Npt- @ptnum)*ch("flatness")*.1);
@up = {0,1,0};
@pscale = (@ptnum/(@Npt - 1.0))*2;
```

SIDE EFFECTS SOFTWARE

# Dedicated Text Editor (first diversion from lecture)

**As your code gets longer you might want a dedicated Text Editor**

**For Windows and Linux**

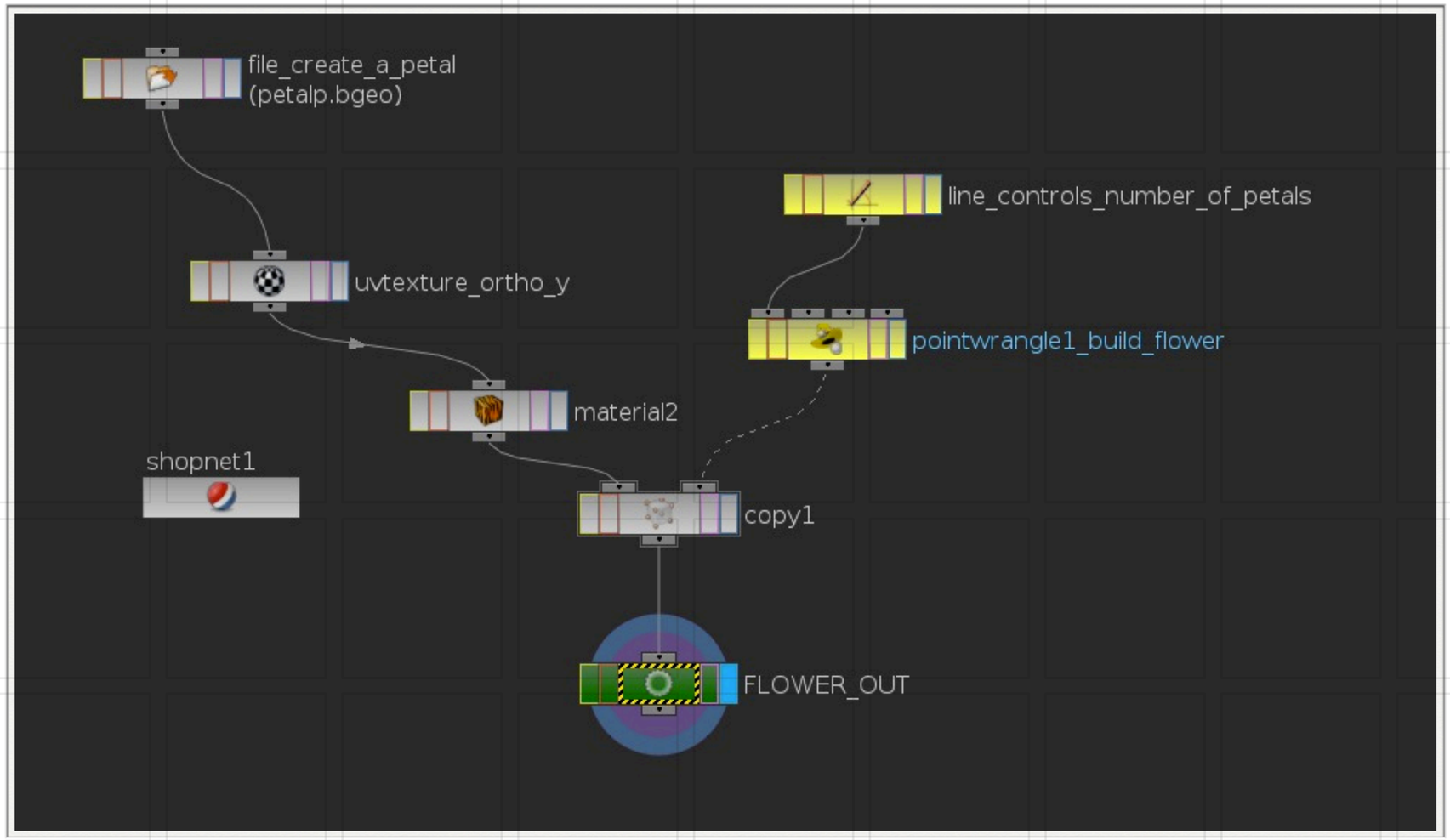
- Cutter - http://www.fundza.com ( I have no personal experience with Cutter, recommended by others)
- Cutter integration into Houdini - http://www.fundza.com/cutter/houdini/about.html

**For Mac OSX**

- TextWrangler - http://barebones.com/products/textwrangler/

SIDE EFFECTS SOFTWARE

# Putting it all together




SIDE EFFECTS SOFTWARE

# Book Used to get Equations


- Flash Math Creativity, Second Edition 2nd edition
- Author - Peters & Tan
- ISBN13 - 978-1590594292

Good book when it was written in 2004.
If you can get it very cheap it is worth it.
Else, I would pass.

SIDE EFFECTS SOFTWARE

# What if you wanted to  use an attribute as a local variable?

By default when you create a variable in a Wrangle node an attribute is created <--- You knew that...

Unfortunately it does not map to a local variable by default

VEX Functions to the rescue

**void addvariablename(string aname, string vname)**

Adds the mapping of the attribute aname to the local variable vname.

Adds the mapping of the attribute aname to the local variable vname. In SOPs that support this, such as the Point SOP, one will then have the local variable $vname referencing the attribute aname. This emulates the behavior of the AttribCreate SOP.

SIDE EFFECTS SOFTWARE

# Example of addvariablename

```
f@G = (1/1.618033989) +ch("pattern_offset"); // The Golden Ratio in Radians
f@rad = ch("radius");
@rad *= ch("growth");
f@rot = @G * (@ptnum);
addvariablename(“rot”,”ROT”); // now I can use  $ROT for instance in a Stamp
@P.x +=  @rad*cos(@rot)*@ptnum*.001;
@P.z +=  @rad*sin(@rot)*@ptnum*.001;
@N.x= @P.x;
@N.z = @P.z;
@N.y = @P.y * ((@Npt- @ptnum)*ch("flatness")*.1);
@up = {0,1,0};
@pscale = (@ptnum/(@Npt - 1.0))*2;
```

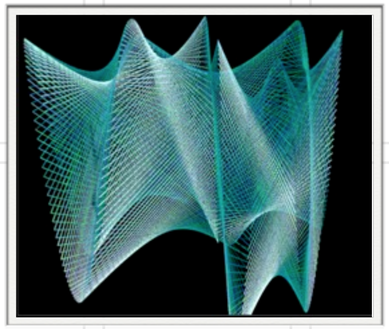
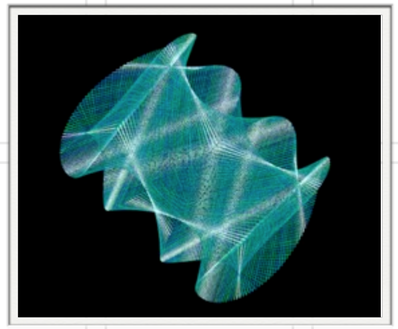
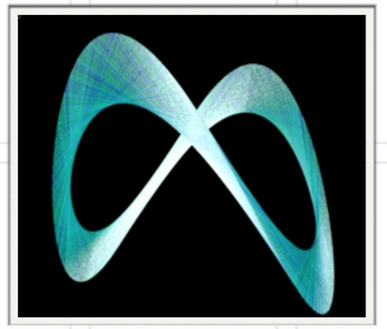
SIDE EFFECTS SOFTWARE

# Class Assignment

Lissajous Figures

SIDE EFFECTS SOFTWARE

# Build Me This

(Physics / General Physics) a curve traced out by a point that undergoes two simple harmonic motions in mutually perpendicular directions. The shape of these curves is characteristic of the relative phases and frequencies of the motion; they are used to determine the frequencies and phases of alternating voltages
[named after Jules A. Lissajous (1822-80), French physicist]

SIDE EFFECTS SOFTWARE

# Book Used to get Equations


- Generative Design - Visualize, Program, and Create with Processsing
- Author - Hartmut Bohnacker, Benedikt Gross, Julia Laub
- ISBN13 - 978-1616890773

Great Book!  Expensive, but worth it.

SIDE EFFECTS SOFTWARE

# Equations You Will Need to Know..

```
x = sin(angle* freqX + radians(phiX)) + cos(angle*freqX);
y = sin(angle* freqY + radians(phiY)) + cos(angle*freqY");
z = sin(angle* freqZ + radians(phiZ));
```

To set an individual component of P

```
P.x = 3;
P.y = 5;
P.z = 7.2;
```

SIDE EFFECTS SOFTWARE

# Solution

```
@P.x = sin(ptnum* ch("freqX") + radians(ch("offsetX"))) + cos(ptnum*ch("freqX"));
@P.y = sin(ptnum* ch("freqY") + radians(ch("offsetY"))) + cos(ptnum*ch("freqY"));
@P.z = sin(ptnum* ch("freqZ") + ch("offsetZ"));
@width = 0.002;
Cd.r = ptnum/(Npt+1);
Cd.g = random(ptnum*8125);
Cd.b = 1.0 - random(ptnum*1234);
i@grpNum = floor(@ptnum/3.0); // here is an example of sorting every three points into a group
```

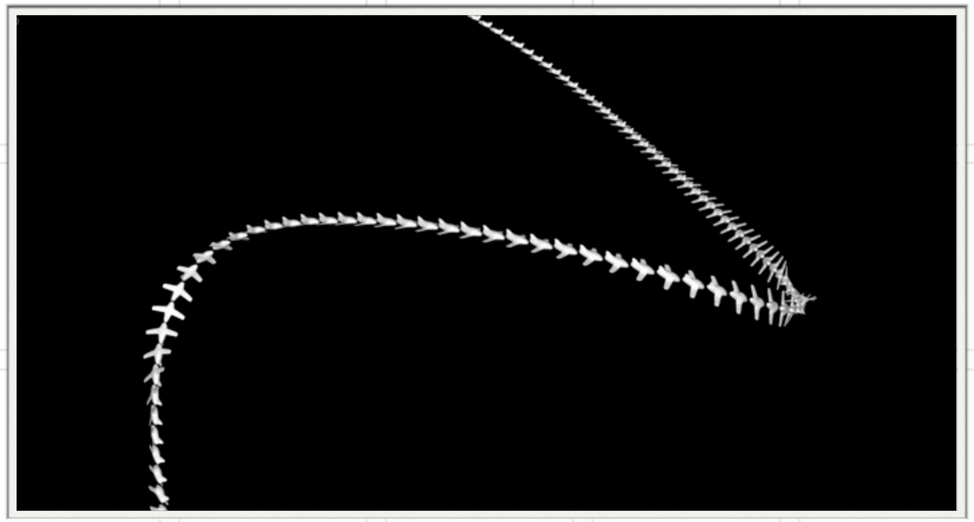
SIDE EFFECTS SOFTWARE

# Conditionals

If..Then Statements

SIDE EFFECTS SOFTWARE

# Objective



- Learn “if then else” statements
- Create Banking Controls for a Path

SIDE EFFECTS SOFTWARE

# If..then..else

As in C and many other languages, you can enclose multiple statements inside curly braces to act as a block.

For example, the if statement can execute one statement:

- if ( needs_zapping() ) zap()

...or a block inside curly braces:

- if ( needs_zapping() ) {
- zap()
- disintegrate()
- remove_dust()
- }

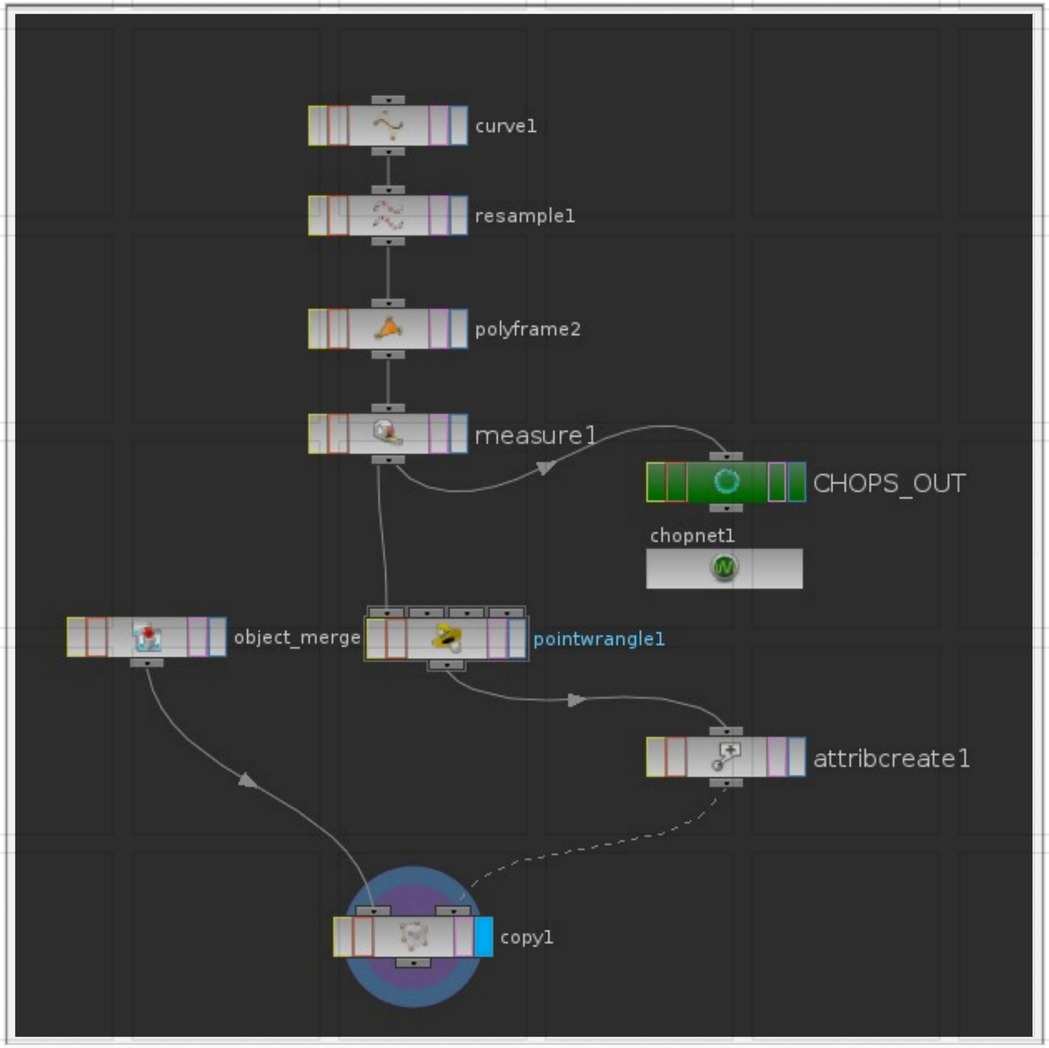
SIDE EFFECTS SOFTWARE

# If..then..else

- if
- if (condition) statement_if_true [else statement_if_false]
- Executes statement_if_true if condition is true.
- If the else clause is included, statement_if_false is executed if condition is false.

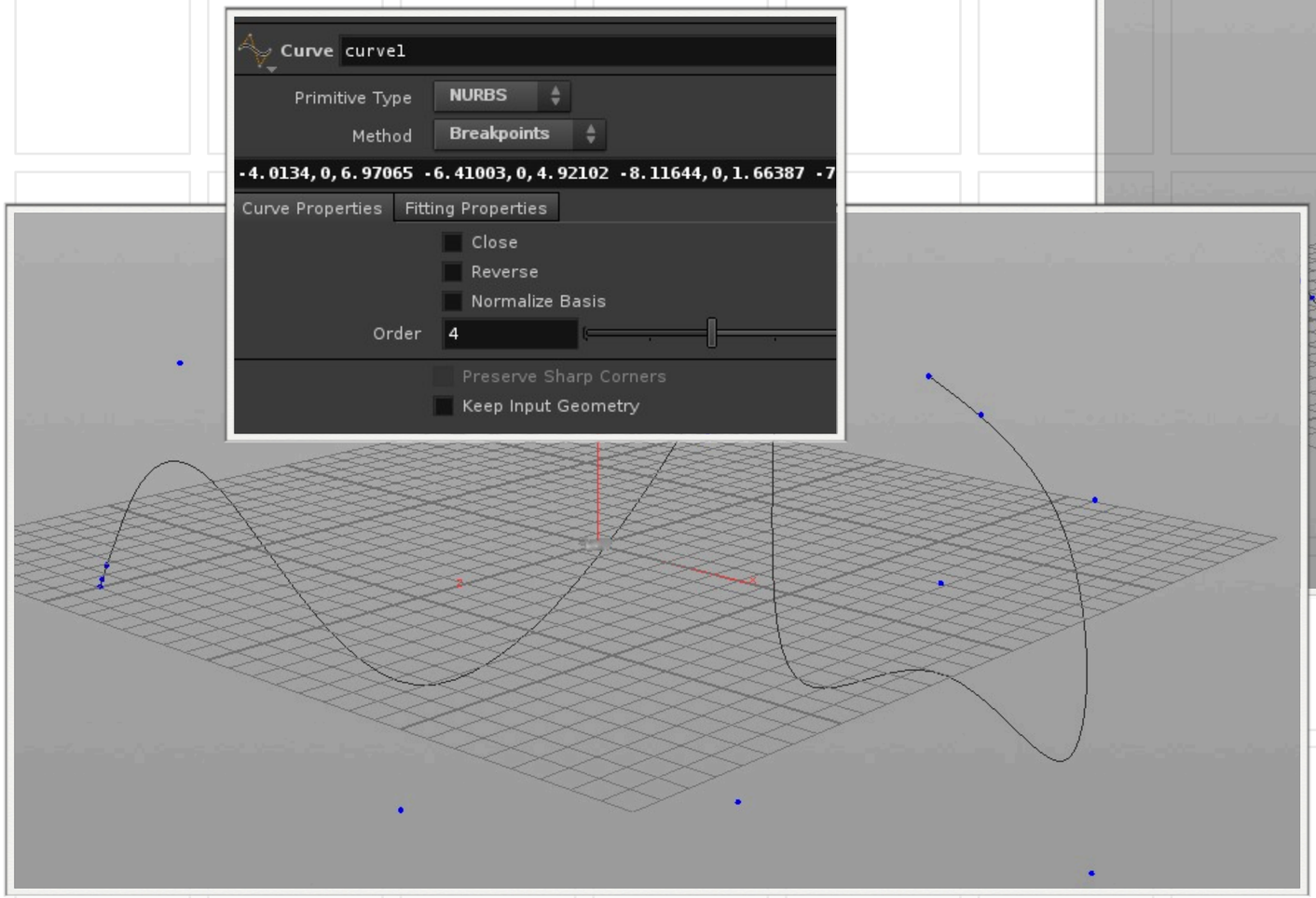
SIDE EFFECTS SOFTWARE

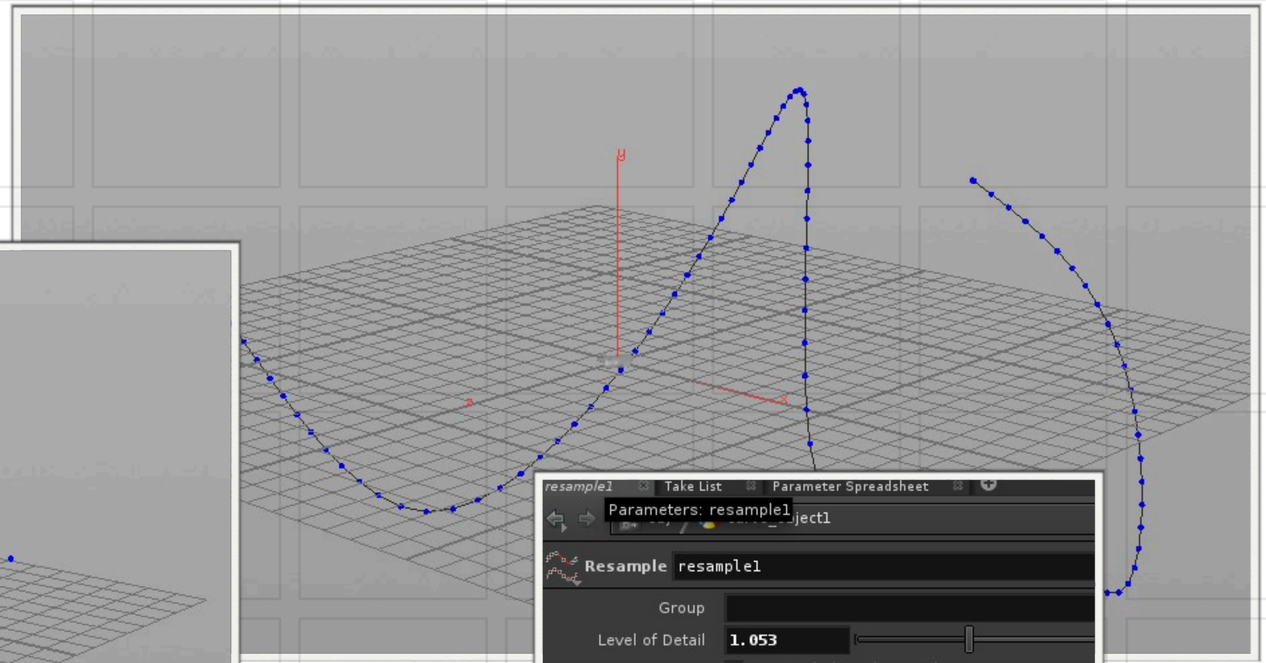
# Going Through the Network




SIDE EFFECTS SOFTWARE

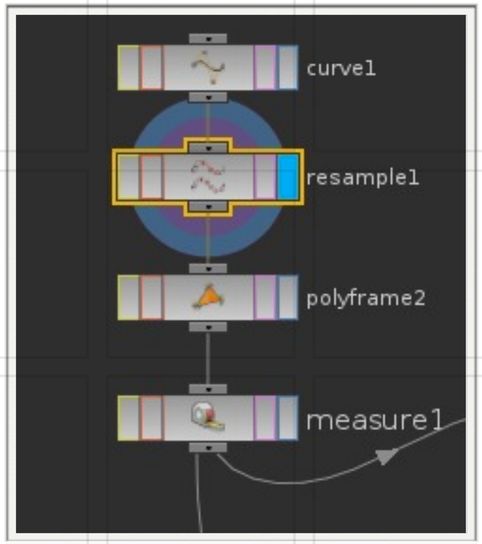
# Create the Curve and Resample

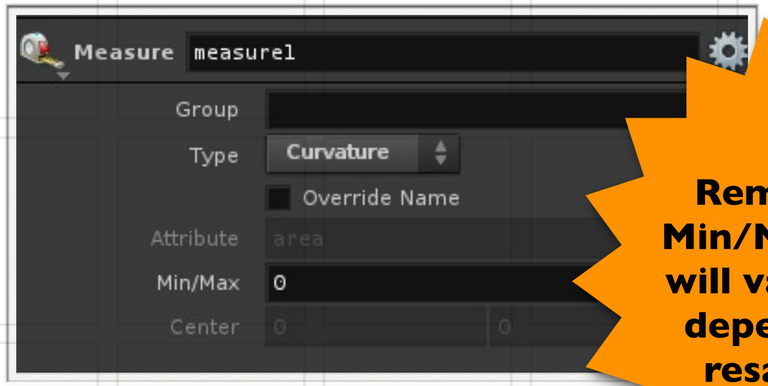


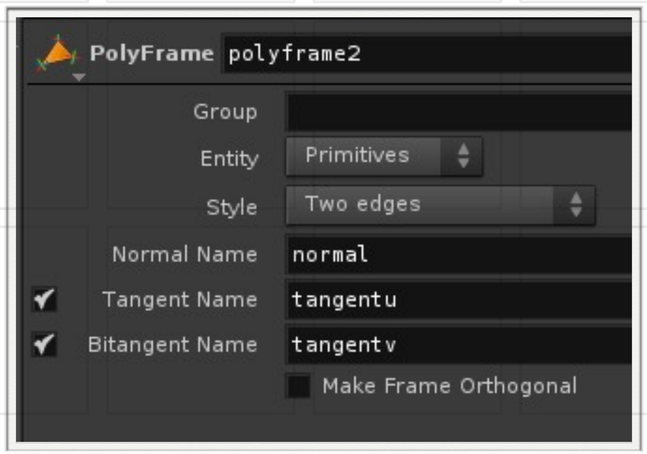

SIDE EFFECTS SOFTWARE

# Polyframe and Measure Curvature



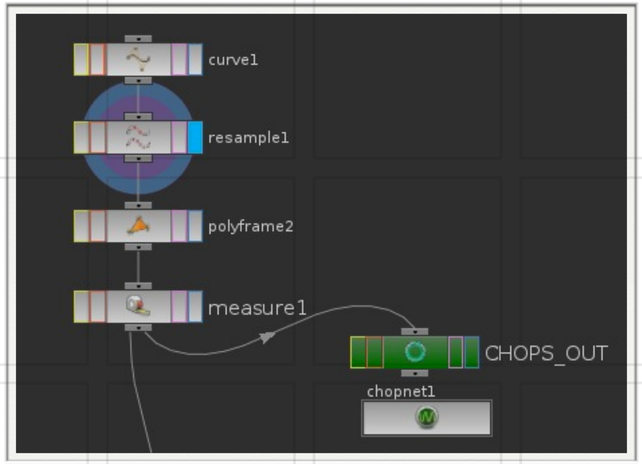

**Remember your Min/Max Curvature will vary quite a bit depending on the resample Value**

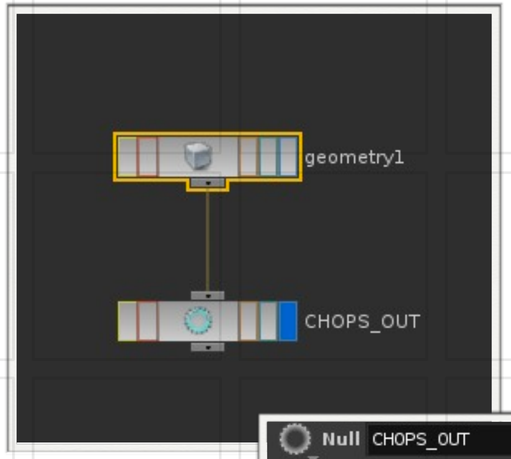


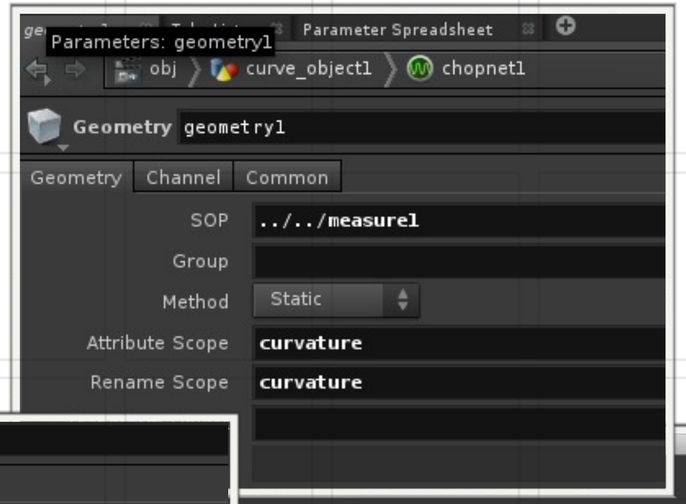

**We can ise CHOPs to automatically calculate Min/Max Curvature**

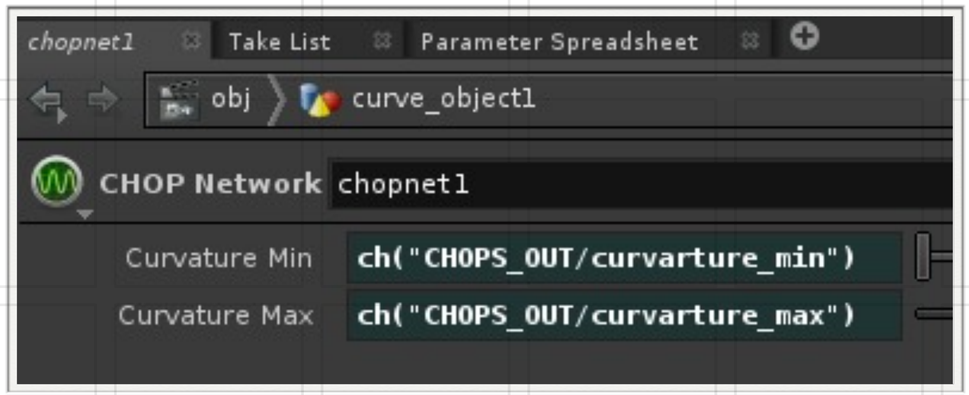
SIDE EFFECTS SOFTWARE

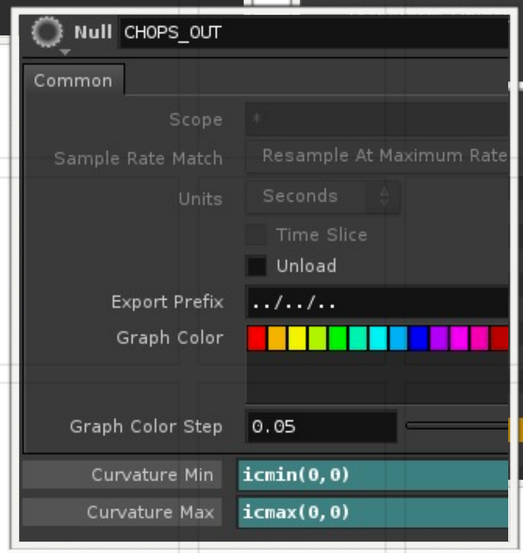
# Setting Up the CHOPNET

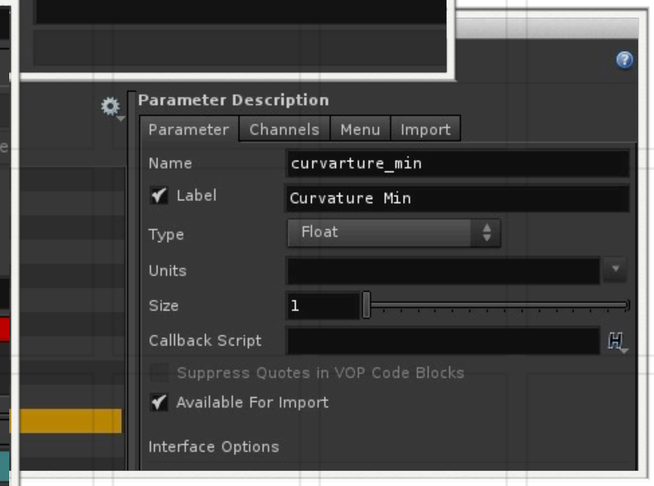


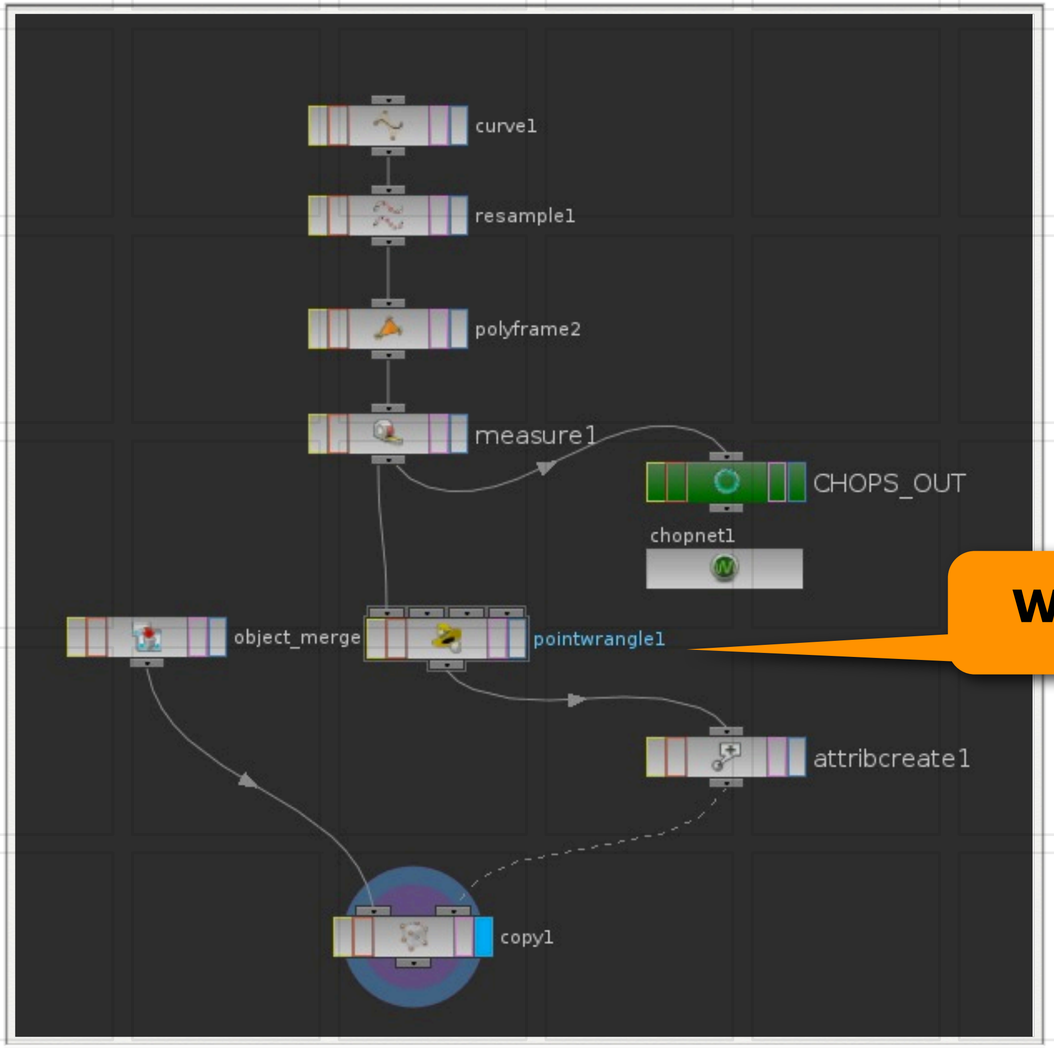




SIDE EFFECTS SOFTWARE

# Time to Write the Point Wrangle

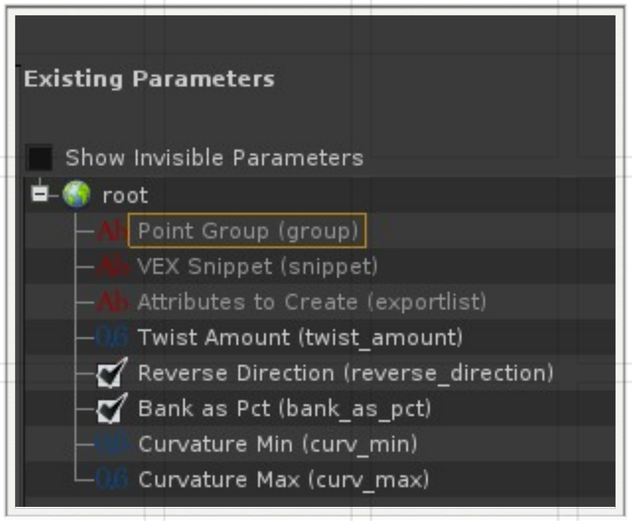
SIDE EFFECTS SOFTWARE

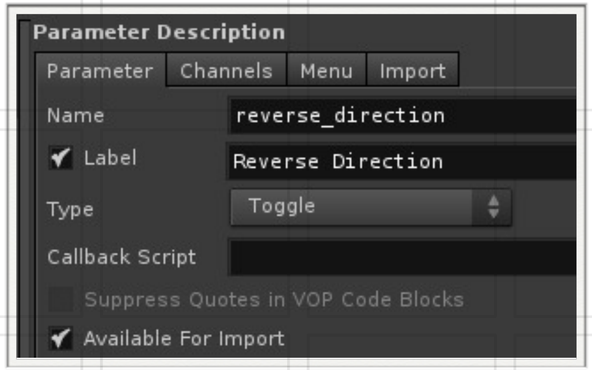
# Going Through the Network

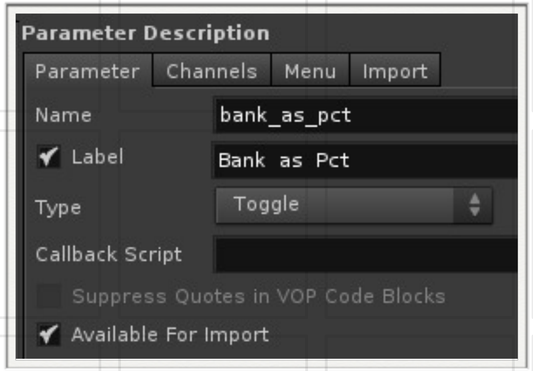


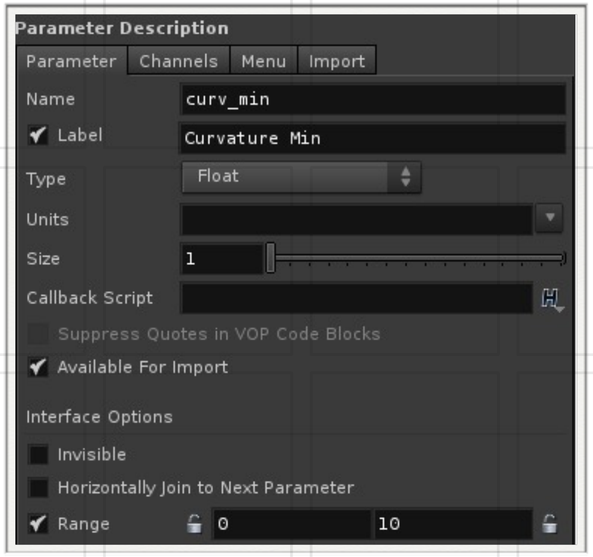

SIDE EFFECTS SOFTWARE

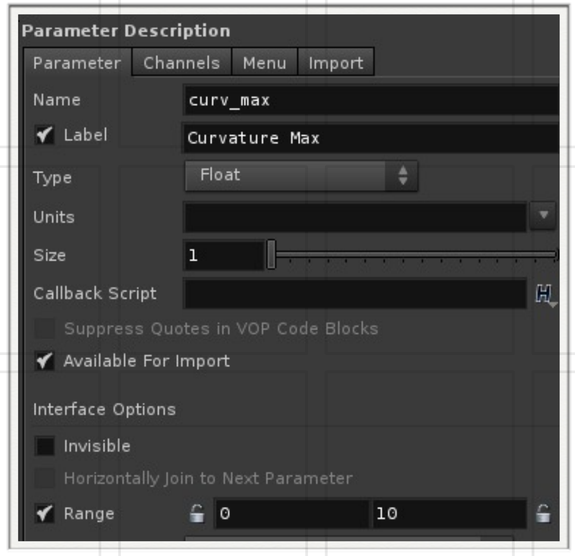
# User Interface



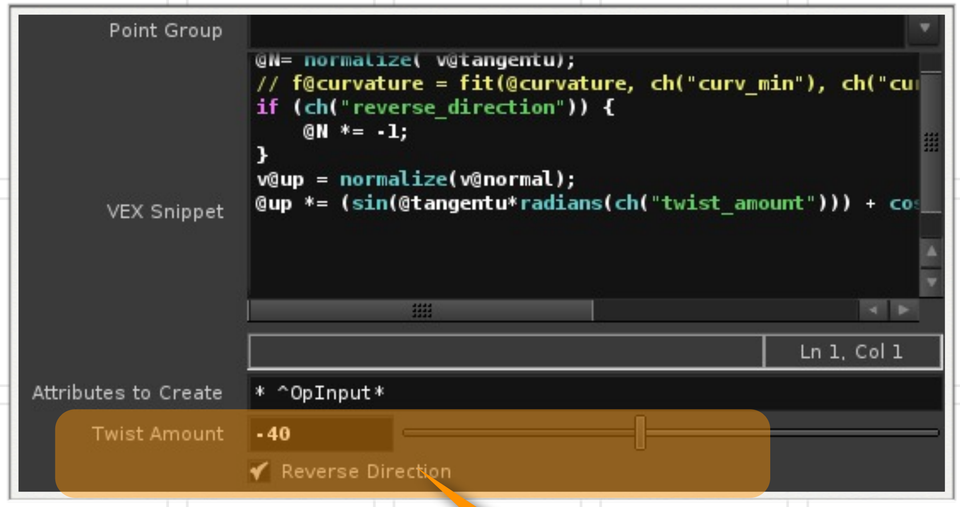




SIDE EFFECTS SOFTWARE

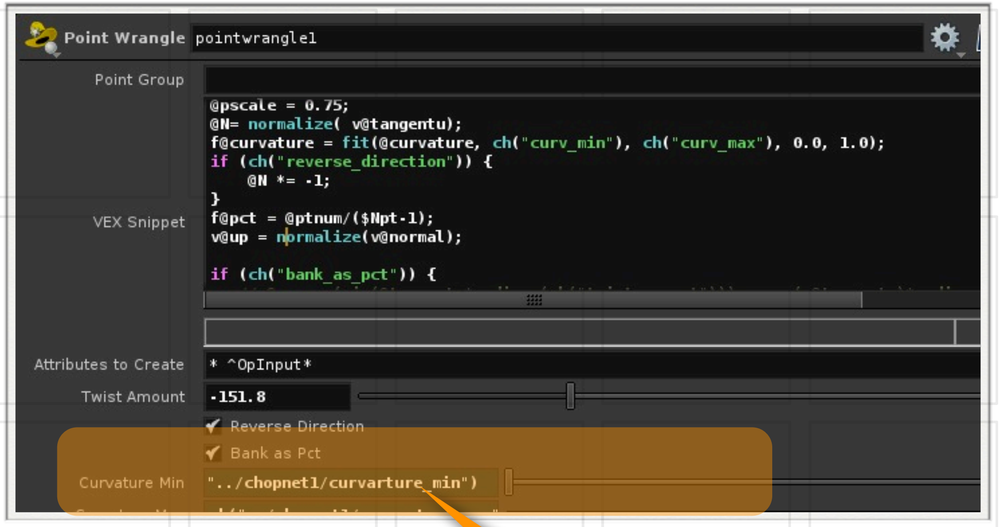
# Simple If Example - Reverse Direction

```
@pscale = 0.75;
@N= normalize( v@tangentu);
if (ch("reverse_direction")) {
   @N *= -1;
}
```

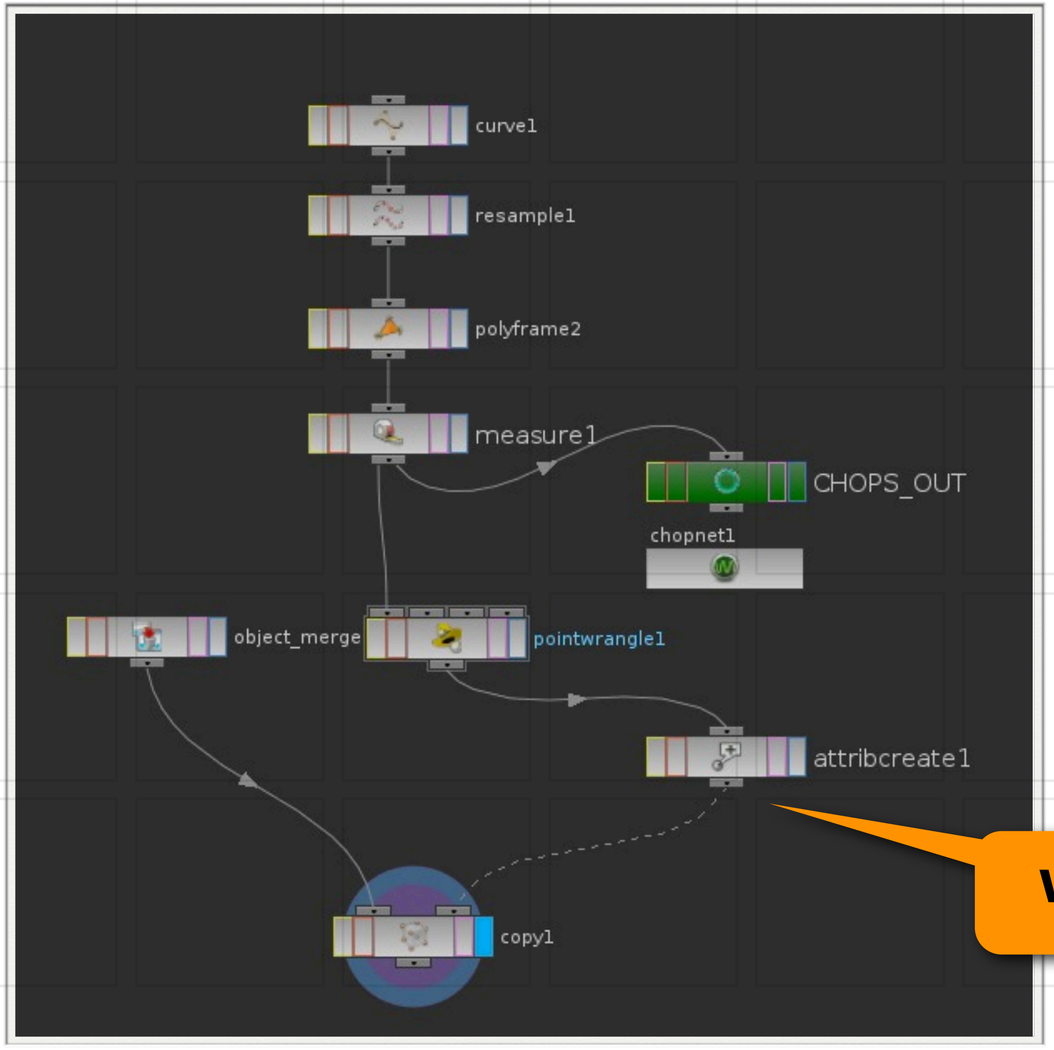


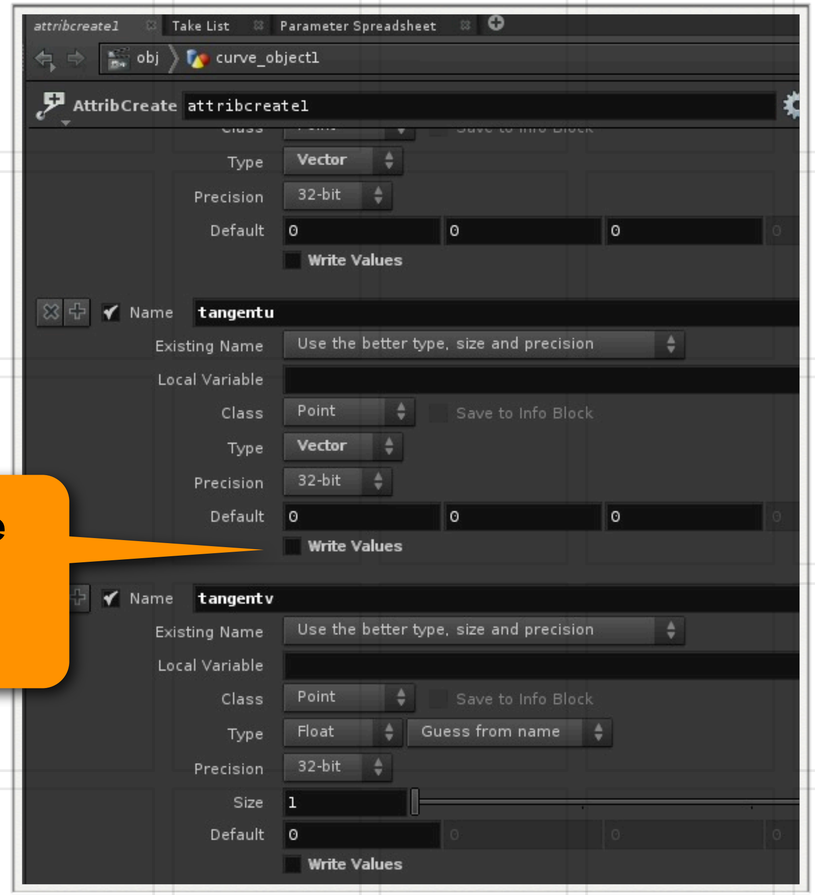
# Simple If ..Else Example - Reverse Direction

```
@pscale = 0.75;
@N= normalize( v@tangentu);
f@curvature = fit(@curvature, ch("curv_min"), ch("curv_max"), 0.0, 1.0);
if (ch("reverse_direction")) {
   @N *= -1;
}
f@pct = @ptnum/($Npt-1);
v@up = normalize(v@normal);
if (ch("bank_as_pct")) {
   @up += (sin(@tangentu*radians(ch("twist_amount"))) + cos(v@tangentv)*radians(ch("twist_amount")))*@pct;
    @up *= (sin(@tangentu*radians(ch("twist_amount"))) + cos(v@tangentv)*radians(ch("twist_amount")))*@curvature;
} else {
   @up *= (sin(@tangentu*radians(ch("twist_amount"))) + cos(@tangentv)*radians(ch("twist_amount")));
}
```



# Binding tangentu & tangentv to Local Variables




**Notice Write value is Deselected**

**We are here!**

SIDE EFFECTS SOFTWARE

# End of Learning VEX and the New Wrangle SOPs

SIDE EFFECTS SOFTWARE